

This invention relates to an asynchronous transfer mode system for, and method of, transferring cells using a control queue in a control memory on one side of a system bus and a status queue in a host memory on the other side of the system bus.

#### RELATED CASES

This application is a continuation of United States patent application 08/764,692, which is hereby incorporated by reference into this application.

#### BACKGROUND OF THE INVENTION

Telephone systems in the United States provide central offices for receiving signals from calling telephones within a particular radius such as one (1) to two (2) miles from the central office and for transmitting telephone signals to such telephones. The telephone signals from a calling telephone are then transmitted through long distances from such central office. The telephone signals then pass to a receiving telephone through a central office within a radius of one (1) mile to two (2) miles from such central office.

The telephone signals are transmitted long distances between central offices through optical fibers which have replaced other media previously provided for such purposes. The optical fibers have certain distinctive advantages over the lines previously provided. They allow a significantly increased number of signals from different telephones to be transmitted at the same time through the optical fibers. They pass the digitally-encoded signals with a higher accuracy than other media.

Various systems have been adopted to carry digitally-encoded signals for telephone, video, and data services. One of such systems now being adopted is designated as asynchronous transfer mode (ATM). This system is

1        advantageous because it recognizes that generally signals  
2        travel in only one direction at any one time between a calling  
3        subscriber and a receiving subscriber. The system preserves  
4        bandwidth in the other direction so that a maximum number of  
5        different messages can be transmitted in such direction.

6  
7                In the prior art, when passing data cells from a  
8        control memory at a first station to a host through a system  
9        bus and then from the host to a recipient, the host remained  
10       on the system bus during all of such transfer. This was  
11       disadvantageous because the recipient received information  
12       from the host only intermittently. During the time that the  
13       recipient did not receive data cells from the host, the system  
14       bus was still connected to the host so that the system bus  
15       could not be used to transfer data cells for other purposes.  
16       As a practical matter, the system bus was tied to the host  
17       about seventy percent (70%) of the time. This allowed the  
18       system bus relatively little time to perform other functions.

19  
20                BRIEF DESCRIPTION OF THE INVENTION

21  
22                In one embodiment of the invention, a status queue  
23       in a host and a control queue in a segmentation and reassembly  
24       (SAR) subsystem are on opposite sides of a host bus in a  
25       control plane. Buffer descriptors in the host and the SAR and  
26       buffers in the host are in a data plane. To transfer cell  
27       payloads to a first line interfacing the SAR, the host writes  
28       the SAR that it has such cell payloads. The host writes the  
29       host buffer descriptors into the control queue to obtain the  
30       transfer of the buffer payload to the first line. The SAR  
31       writes the status queue when the transfer has been completed.

1 To transfer cell payloads to the host memory, the host writes  
2 into the control queue the address of the buffers to receive  
3 the payload from the SAR. The SAR then writes the buffer  
4 descriptors to the status queue to obtain the transfer of the  
5 cell payloads to the buffers. Each of the control and status  
6 queues may be respectively considered to constitute two (2)  
7 control queues and two (2) status queues. The SAR determines  
8 if either status queue is full by comparing the address  
9 written by the SAR into such status queue with the address  
10 written by the SAR periodically to the SAR where the host is  
11 in the status queue. The host determines if either control  
12 queue is full by comparing the address written by the host  
13 into such control queue with the address written by the SAR  
14 periodically to the host where the SAR is in the control  
15 queue.

#### 17 BRIEF DESCRIPTION OF THE DRAWINGS

19 In the drawings:

20 Figure 1 is a schematic block diagram illustrating  
21 on a simplified basis the paths for transferring cells of  
22 information in an asynchronous transfer mode between a calling  
23 station and a receiving station through an access multiplex;

24 Figure 2 is a schematic block diagram on a  
25 simplified basis of a system included in the system shown in  
26 Figure 1 for transferring cell payloads to a transmit cell  
27 interface from a host memory or for transferring cell payloads  
28 to the host memory from a receive cell interface;

29 Figure 3 is a schematic block diagram showing in  
30 additional detail the operation of the sub-system shown in  
31 Figure 2 when the cell payloads are transferred from the  
32 receive cell interface to the host memory;

1           Figure 4 is a schematic block diagram showing in  
2 additional detail the operation of the sub-system shown in  
3 Figure 2 when the cell payloads are transferred from the host  
4 memory to the transmit cell interface; and

5           Figure 5 is a schematic diagram indicating the  
6 operation of stages in control and data planes when  
7 information in a host memory on one side of a system bus is to  
8 be transferred to the transmit cell interface (shown in Figure  
9 2) on the other side of the system bus;

10          Figure 6 is a schematic diagram indicating the  
11 operation of stages (also shown in Figure 5) in the control  
12 and data planes when information is to be transferred from the  
13 receive cell interface (shown in Figure 2) on one side of the  
14 system bus to the host memory on the other side of the system  
15 bus;

16          Figures 7 and 8 provide flow charts indicating the  
17 process of a control queue (shown in Figures 5 and 6) during  
18 the operations of segmentation and re-assembly;

19          Figures 9 and 10 provide flow charts indicating the  
20 process of a status queue (shown in Figures 5 and 6) during  
21 the operations of segmentation and re-assembly;

22          Figure 11 is a table providing a definition of  
23 certain terms used in the flow charts shown in Figures 7 and  
24 8; and

25          Figure 12 is a table providing a definition of  
26 certain terms used in the flow charts shown in Figures 9 and  
27 10.

#### 28 29           DETAILED DESCRIPTION OF THE INVENTION 30

31          Figure 1 illustrates in block form a system  
32 generally indicated at 10 and known in the prior art for

transferring signals to and from a pair of telephones (or sources) 12 and 14 respectively through lines 16 and 18 to a common access multiplex 20. The telephone (or source) 12 may illustratively transmit or receive television signals and telephone (voice) signals on a line 16 and the telephone (or source) 14 may illustratively transmit or receive television signals and telephone (voice) signals on a line 18. All signals are digitally encoded. For purposes of simplification, the television signals are shown in Figure 1 as being transferred in solid lines and the telephone signals are shown in Figure 1 as being transferred in broken lines.

The signals in the lines 16 and 18 pass to the access multiplex 20. The respective digitally-encoded transmit signals are segmented into fixed-length cell payloads and a cell header is added to each cell payload to form a cell. Similarly, received cells are reassembled into the respective receive cells. The headers of the cells are generated in the access multiplex to provide a virtual channel indication and/or a virtual path indication. The header indicates the path which is being followed to pass the cells to a central office 22. The central office 22 may modify the header again in the cells to identify the path through which the cells are subsequently being transferred. The cells may then be transferred either to a television access 24 or to a telephone access 26 at receiving stations generally indicated at 28 in Figure 1.

Figure 2 illustrates one embodiment of a sub-system generally indicated at 29 and enclosed within a rectangle defined by broken lines for use with the access multiplex 20 shown in Figure 1 for providing a controlled transfer of ATM

1 cell payloads between a line 30 from a receive cell interface  
2 and a host memory 32. When the cells are transferred from the  
3 line 30, the cells pass through a receive FIFO 34. The FIFO  
4 34 constitutes a first-in-first-out memory well known in the  
5 art to provide a time buffer. The payload in each cell then  
6 passes to a reassembly direct memory access (DMA) stage 36.  
7 The header in each cell passes to a reassembly state machine  
8 40 for processing.

9  
10 The header in each cell is introduced from the  
11 reassembly state machine 40 to a control memory 38 which  
12 processes the header to provide addresses that indicate where  
13 the cell payloads are to be stored in the host memory 32. The  
14 addresses are then applied through the reassembly state  
15 machine 40 to the reassembly direct memory access (DMA) stage  
16 36 to direct the payload from the FIFO 34 through a host  
17 interface 42 to a host bus 44. The cells are then transferred  
18 in the host memory 32 to the addresses indicated by the  
19 control memory 38.

20  
21 Cells may also be transferred to a transmit cell  
22 interface through a line 45 by the sub-system 29 shown in  
23 Figure 2. The segmentation state machine 50 reads addresses  
24 from the control memory 38 that indicate where cell payloads  
25 are stored in the host memory 32. The addresses are then  
26 applied by the segmentation state machine 50 to the  
27 segmentation direct memory access (DMA) 46 to direct the cell  
28 payloads to the transmit FIFO 48. The transmit FIFO 48 may be  
29 constructed in a manner similar to the receive FIFO 34. The  
30 header is introduced by the control memory 38 to the  
31 segmentation state machine 50 for combination in the transmit  
32

1 FIFO 48 with the payload. The recombined cell then passes to  
2 the transmit cell interface line 45.

3  
4 Figure 3 illustrates in additional detail the  
5 operation of the sub-system shown in Figure 2 in separating  
6 the header and the payload in a cell, reassembling the cell  
7 payloads and recording the reassembled payloads in the host  
8 memory 32. In the flow chart shown in Figure 3, the cell  
9 header is initially read as at 70. The header is used to  
10 compute a "connection index" (see block 72) to yield a memory  
11 address in a reassembly state. This is indicated as a table  
12 73 designated as "Reassembly State" in the control memory 38.  
13 The table 73 contains a plurality of virtual channel  
14 connections which are respectively designated as "VCC 1", "VCC  
15 2", "VCC 3", etc.

16  
17 Each of the virtual channel connections contains a  
18 table 75 which provides certain information including the  
19 address of a region of the host memory 38, the length of the  
20 region in the host memory and the protocol information for the  
21 virtual channel connection VCC. Figure 3 schematically shows  
22 that the table containing the address region in the host  
23 memory 38, the length of the region and the protocol  
24 information for the virtual channel connection VCC are being  
25 selected from the virtual channel connection designated as  
26 "VCC 2". This is indicated by broken lines at 74 and by the  
27 table 75 in Figure 3. It will be appreciated that this is  
28 schematic and illustrative and that other VCC's may be  
29 selected.

30  
31 The cell from the line 30 in Figure 2 relating to  
32 the receive cell interface is then checked with the protocol

1 information in the VCC 2 virtual channel connection in the  
2 table 75 in the control memory 38 as indicated at 76 in Figure  
3 3. If the check indicates that the protocol information in  
4 the header and the payload is correct, the region address in  
5 the host memory 32 and the length of such region are read from  
6 the VCC 2 block in the control memory 38 as indicated at 78 in  
7 Figure 3. The region address in the host memory 32 is passed  
8 to the reassembly DMA 36 in Figure 2 as indicated at 80. The  
9 reassembly DMA 36 is then activated to transfer the cell  
10 payload from the receive FIFO 34 in Figure 2 to the host  
11 memory 32 as indicated at 82 in Figure 3.

12  
13 As the successive cell payloads for the VCC 2 table  
14 73 are reassembled in the region, a check is made in each  
15 reassembly to determine if the end of the region in the VCC 2  
16 channel connection has been reached. This is indicated at 84  
17 in Figure 3. If the answer is "No", the region address for  
18 successive cells is incremented for the successive payloads in  
19 the VCC 2 channel connection recorded in the host memory  
20 region and the region length is decremented by the same  
21 amount. A block 86 in Figure 3 indicates this.

22  
23 If the end of the region in the VCC 2 table in the  
24 control memory 38 has been reached, a "Yes" indication is  
25 provided from the block 84. This causes a block 88 to be  
26 activated in Figure 3. This block is designated as "Read Free  
27 Region". The control memory 38 contains a Free Region Queue  
28 indicated at 90 in Figure 3. When the block 88 is activated,  
29 it causes the next entry in the Free Region Queue 90 to be  
30 selected. For example, when entry 1 in the Free Region Queue  
31 has been previously selected, entry 2 in the Free Region Queue  
32 90 is now selected. This is indicated by broken lines 92



1 extending from the entry 2 in the Free Region Queue 90 to a  
2 table 94 in Figure 3.

3  
4 Entry 2 in the Free Region Queue contains a new  
5 region address in the host memory 38 and the length of such  
6 region. This information is transferred to the table 75 in  
7 place of the information previously recorded in the table.  
8 The blocks 78, 80, 82, 84, 86 and 88 are now operated as  
9 discussed above to transfer the payloads in the cells on the  
10 line 30 to the regions in the host memory 32. At the end of  
11 this region, entry 3 in the Free Region Queue may be selected  
12 to provide a new region address in the host memory 32 and the  
13 length of such region if the payload has not been completely  
14 recorded in the host memory 32. The steps described above are  
15 repeated in this manner until all of the payload has been  
16 recorded in the host memory 32.

17  
18 Figure 4 indicates in additional detail the  
19 operation of the sub-system shown in Figure 2 in transferring  
20 the cell payloads from the host memory 32, reading the header  
21 from the control memory 38 to indicate the ATM path, combining  
22 the header and the payload into a cell and transferring the  
23 cell to the line 45 in Figure 2. In the flow chart shown in  
24 Figure 4, a block 100 is provided to determine if a VCC cell  
25 has been scheduled for a particular time slot. If a cell has  
26 not been scheduled, an idle cell (i.e. no cell recorded in a  
27 time slot) is transferred to the line 45 in Figure 2. This is  
28 indicated by a line 101 in Figure 4.

29  
30 If a cell has been scheduled for the particular time  
31 slot, the block 100 in Figure 4 selects a virtual channel  
32 connection in a table 102 in the control memory 38. This

1 table is designated as "Segmentation State" in Figure 4. As  
2 shown in Figure 4, the table 102 contains a plurality of  
3 virtual channel connections which are illustratively  
4 designated as "VCC 1", "VCC 2", "VCC 3", etc. The virtual  
5 channel connection VCC 2 is illustratively shown as being  
6 selected in the table 102. This is indicated by broken lines  
7 104. It will be appreciated that any other block could have  
8 been chosen. The virtual channel connection VCC 2  
9 illustratively includes a header value (to indicate the path  
10 of transfer of the cell), a region address, a region length,  
11 protocol information and the position of the next region  
12 description in the host memory. This is illustrated at 106 in  
13 Figure 2.

14  
15 The header value and the protocol information in the  
16 VCC 2 block are read from the control memory 38 as indicated  
17 at 108 and 109 respectively in Figure 4. The header value is  
18 then transferred to the transmit FIFO 48 in Figure 2 as  
19 indicated at 110 in Figure 4 and the region address and length  
20 are read from the VCC 2 virtual channel connection as  
21 indicated at 112 in Figure 4. The segmentation DMA 46 in  
22 Figure 2 is then set up (see block 114 in Figure 4) and the  
23 payload is transferred from the host memory region to the  
24 transmit FIFO 48 in Figure 2 (see block 116 in Figure 4). A  
25 check is made in each transfer of the payload of a successive  
26 cell to determine if the region address being transferred for  
27 the virtual channel connection 106 is at the end of its  
28 length. This is indicated at 118 in Figure 4.

29  
30 If the end of the host region in the VCC 2 virtual  
31 channel connection has not been reached as indicated at 120 in  
32 Figure 4, the region address at 106 in the control memory is

1 incremented to account for the successive payload transferred  
2 to the transmit FIFO 48 and the region length is decremented  
3 by the same amount (see block 122). This provides an updated  
4 record of the region being processed in the virtual channel  
5 connection VCC 2 and an updated record of the remaining length  
6 of the region to be processed in the virtual channel  
7 connection VCC 2.

8  
9 When the end of the region in the virtual channel  
10 connection VCC 2 has been reached, the address of the next  
11 region in the host memory 38 and the length of this region are  
12 read as indicated at 124. This next region is indicated as  
13 "next" in the table 106 and is indicated in additional detail  
14 by a table 128 in Figure 4. The table 128 is designated as a  
15 "Region Descriptor" to conform to the designation in the block  
16 124. The table 128 also contains a block designated as  
17 "Next". The table 128 is then transferred to the position of  
18 the table 106 to replace the information previously in the  
19 table 106. The address information transferred from the table  
20 128 to the table 106 is then processed in the blocks 108, 109,  
21 110, 112, 114, 116, 118, 120, 122 and 124 in the same manner  
22 as described above. Upon the completion of the processing of  
23 the region in the table 106, the "Next" block in the table 106  
24 is processed to determine the subsequent host region address  
25 in the host memory 32 and the length of this region address.

26  
27 Figure 5 is a schematic diagram showing the position  
28 of a system bus such as a PCI bus. This position is indicated  
29 by broken lines 130. The PCI bus may be considered to  
30 correspond to the host bus 44 in Figure 2. A control queue  
31 131 (which may be included in the control memory 38 in Figure  
32 2) in a segmentation and re-assembly (SAR) sub-system

1 corresponding to the sub-system 29 in Figure 2 is shown  
2 schematically in Figure 5 to the right of the position 130 of  
3 the PCI bus. The control queue 131 may be considered to  
4 constitute two (2) control queues,, one as shown in Figure 5  
5 and the other as shown in Figure 6. The control queues 131 as  
6 shown in Figures 5 and 6 may be considered to have  
7 substantially identical constructions.

8  
9 A status queue 132 is shown on the host side of the  
10 PCI bus 130. The status queue 132 may be considered to be  
11 included in the host memory 32 in Figure 2. The status queue  
12 132 may be considered to constitute two (2) status queues, one  
13 as shown in Figure 5 and the other as shown in Figure 6. The  
14 status queues 132 as shown in Figures 5 and 6 may be  
15 considered to have substantially identical constructions.

16  
17 The status queues 132 in Figures 5 and 6 are  
18 associated with a plurality of buffers 134a, 134b, 134c, etc.  
19 on the host side of the PCI bus 130. The buffers 134a, 134b  
20 and 134c may be considered to be included in the host memory  
21 32. A plurality of buffer descriptors 136a, 136b, 136c, etc.  
22 on the host side of the PCI bus 130 are respectively connected  
23 to individual ones of the buffers 134a, 134b and 134c. The  
24 buffer descriptor 136a is connected to the buffer descriptor  
25 136b and the buffer descriptor 136b is connected to the buffer  
26 descriptor 136c. The buffer descriptor 136a has a connection  
27 145a from the status queue 132.

28  
29 The status queue 132 and the control queues 131 are  
30 disposed in a control plane. The buffers 134a, 134b and 134c  
31 and the buffer descriptors 136a, 136b and 136c are disposed in  
32 a data plane. The separation between the control plane and

1 the data plane is indicated by a broken line 138 in Figure 5.  
2 Buffer descriptors 140a, 140b and 140c are disposed in the  
3 data plane on the same side of the PCI bus 130 as the control  
4 queue 131. The buffer descriptors 140a, 140b and 140c are  
5 mirror images of the buffer descriptors 136a, 136b and 136c.  
6 A connection or pointer 145b is provided from the control  
7 queue 131 to the buffer descriptor 140a. The buffer  
8 descriptor 140a is connected to the buffer descriptor 140b and  
9 the buffer descriptor 140b is connected to the buffer  
10 descriptor 140c in a manner similar to the connections between  
11 the buffer descriptors 136a, 136b and 136c.

12  
13 When data cells are to be transferred from the host  
14 memory 32 to the line 45 in Figure 2, the host writes to the  
15 control queue 131 across the PCI bus 130 in Figure 5 that it  
16 has information in the host memory 32 that it wishes to  
17 transfer to the line 45. This is provided by the pointer 145b  
18 from the control queue 131 to the buffer descriptor 140a. The  
19 segmentation and re-assembly (SAR) sub-system 29 then reads  
20 the cell payload from the host memory 32 and transfers this  
21 information to the line 45. After all of the information to  
22 be transferred from the host memory 32 has been transferred to  
23 the line 45, the segmentation and re-assembly (SAR) sub-system  
24 29 writes to the status queue 132 that it has completed the  
25 transfer. This is provided by the pointer 145a to the buffer  
26 descriptor 140a from the status queue 132.

27  
28 During the transfer described in the previous  
29 paragraph, the control queue 131 in Figure 5 initially  
30 activates the buffer descriptor 140a to obtain the transfer of  
31 the cell payload across the PCI bus to the SAR 29 from the  
32 buffer 134a associated with the buffer descriptor 140a. When

1 all of the information has been transferred from the buffer  
2 134a, the buffer descriptor 140a causes the buffer descriptor  
3 140b to be activated. This causes the cell payload to be  
4 transferred across the PCI bus to the SAR 29 from the buffer  
5 134b. In like manner, the cell payload becomes transferred  
6 across the PCI bus 130 from the buffer 134c to the SAR 29 when  
7 all of the cell payload has been transferred from the buffer  
8 134b.

9  
10 As will be appreciated from the above discussion,  
11 the control queue 131 in Figure 5 does not transfer cell  
12 payloads to the segmentation and re-assembly (SAR) sub-system  
13 29. The control queue 131 provides control information to the  
14 buffer descriptors 140a, 140b and 140c in the data plane but  
15 these buffer descriptors are on the same side of the PCI bus  
16 as the control queue. This control information constitutes  
17 pointers to the buffer descriptors 140a, 140b and 140c on the  
18 SAR side. The pointers may be considered as addresses to the  
19 buffer descriptors 140a, 140b and 140c.

20  
21 In response to the control information described in  
22 the previous paragraph, the buffer descriptors 136a, 136b and  
23 136c provide for the transfer of the cell payloads in the  
24 buffers 140a, 140b and 140c across the PCI bus 130 to the SAR  
25 29 for passage to the line 45 in Figure 2. When the transfer  
26 of the cell payload to the line 45 has been completed, the SAR  
27 29 writes to the host that it has completed the transfer. In  
28 this writing, the SAR 29 points to the buffer descriptors  
29 136a, 136b and 136c on the host side. By providing this  
30 arrangement, the PCI bus 130 is not tied up by having the host  
31 read across the PCI bus while the transfer of the cell payload  
32 from the host buffers 134a, 134b and 134c across the PCI bus

1 130 to the SAR 29 is taking place.

2  
3 As will be seen, buffers (134a, 134b and 134c) are  
4 provided only on the host side of the PCI bus 130. However,  
5 buffer descriptors (136a, 136b and 136c and 140a, 140b and  
6 140c) are provided on both sides of the PCI bus 130. The  
7 buffer descriptors 140a, 140b and 140c are provided on the SAR  
8 side of the PCI bus 130 so that the SAR 29 will not have to  
9 read across the PCI bus 130. The SAR 29 is able to read the  
10 buffer descriptors 140a, 140b and 140c on the SAR side of the  
11 PCI bus 130 and provide the indications to the host that all  
12 of the cell payload in the buffers 134a, 134b and 134c has  
13 been transferred from the host to the line 45.  
14

15 A read pointer 148 is provided in the host to the  
16 status queue 132. This pointer is designated as "READ". This  
17 pointer indicates the address in the status queue 132 in  
18 Figure 5 where information will be read out of the status  
19 queue at any instant. A pointer 162 designated as "READ\_UD"  
20 (meaning read-update) extends from the status queue 132 across  
21 the PCI bus to the SAR 29. This pointer is essentially a copy  
22 of the pointer 148. This pointer is provided so that the SAR  
23 29 will not have to read across the PCI bus 130 to ascertain  
24 the address read by the read pointer 148 at the host.  
25

26 As indicated by the pointer 162, the host writes  
27 periodically across the PCI bus 130 to the SAR 29 the address  
28 where information is being read by the host out of the status  
29 queue 132. This writing occurs only periodically to minimize  
30 the amount of time that the PCI bus 130 is tied up in the  
31 writing of such address from the host to the SAR 29. A  
32 pointer 164 is also provided from the SAR across the PCI bus

1 130 to the status queue 132. This pointer is designated as  
2 "WRITE". It provides an indication from the SAR of the  
3 address in the status queue 132 in Figure 5 where the SAR is  
4 going to write next into the status queue. The pointers 148,  
5 162 and 164 are also shown in Figure 6.

6  
7 In Figure 5, a pointer is indicated at 160 and is  
8 designated as "READ". It provides an address indicating where  
9 the SAR 29 is going to read the control queue 131 in Figure 5  
10 at any instant. A pointer 150 extends from the host to the  
11 control queue 131 in Figure 5. This pointer is designated as  
12 "READ\_UD" (meaning read update). This pointer is essentially  
13 a copy of the pointer 160. As indicated by this pointer, the  
14 SAR writes periodically across the PCI bus 130 to the host the  
15 address where the information is being read out of the control  
16 queue 131. This writing is provided so that the host will not  
17 have to read across the PCI bus to ascertain the address read  
18 by the pointer 160 in the SAR. The writing provided by the  
19 pointer 150 occurs only periodically to minimize the amount of  
20 time that the PCI bus 130 is tied up in the writing of such  
21 address from the SAR. A pointer 152 is also provided from the  
22 host across the PCI bus 130 to the control queue 131 in Figure  
23 5. This pointer is designated as "WRITE". It indicates the  
24 address in the control queue 131 where the host is going to  
25 write next into the control queue. The pointers 160, 150 and  
26 152 are also indicated in Figure 6.

27  
28 In Figure 5, the host writes into the control queue  
29 131 information that points to full buffers 134a, 134b and  
30 134c. This pointer indicates that information can be  
31 transferred out of the buffers 134a, 134b and 134c across the  
32 PCI bus 130 to the SAR 29 and then to the line 45 in Figure 2.



1 In Figure 5, the SAR 29 writes across the PCI bus 130 into the  
2 status queue 132, pointing to the empty buffers 134a, 134b and  
3 134c. This indicates that all of the cell payload in these  
4 buffers has been transferred out of these buffers across the  
5 PCI bus 130 to the line 45.

6  
7 Figure 6 has the same stages in the control plane as  
8 Figure 5. However, in Figure 6, cell payloads are being  
9 transferred in the data plane from the interfacing line 30 in  
10 Figure 2 to the buffers 134a, 134b and 134c in the host memory  
11 32. Because of this, there are some differences in the data  
12 plane between Figures 5 and 6. In Figure 6, the host writes  
13 across the PCI bus 130 to the SAR 29 information that points  
14 to the empty buffers 134a, 134b and 134c. This is indicated  
15 by pointers 170 extending between the buffer descriptors 136a,  
16 136b and 136c and the control queue 131 in Figure 6. These  
17 pointers indicate to the SAR 29 that the cell payload from the  
18 interfacing line 30 in Figure 2 can be transferred across the  
19 PCI bus 130 to the buffers 134a, 134b and 134c in the host.  
20 In Figure 6, the SAR writes across the PCI bus 130 into the  
21 status queue 132, pointing to full buffers 134a, 134b and 134c  
22 in the host. In effect, the SAR 29 is pointing to the buffers  
23 134a, 134b and 134c so that the host can transfer the cell  
24 payload in the buffers to sources connected to the host.

25  
26 The word "Circular" is used in Figures 5 and 6 in  
27 modification of the words "Status Queue" to identify a  
28 preferable type of status queue 132. The word "circular" is  
29 also used in Figures 5 and 6 in modification of the words  
30 "control queue" to identify a preferable type of control queue  
31 131.

1            Figures 7 and 8 provide a flow chart indicating the  
2            process of the control queue 131 during segmentation and re-  
3            assembly. Figure 7 relates to the control queue processing of  
4            host flow and Figure 8 relates to the control queue processing  
5            of SAR flow. In reading the successive blocks in Figure 7,  
6            reference should be made to the table in Figure 11 which  
7            provides a definition of the different terms shown in the  
8            different blocks in Figure 7. These definitions are  
9            supplemented by the previous discussion relating to Figures 5  
10           and 6 and the subsequent discussion relating to Figures 7 and  
11           8.

12  
13           The functions shown in Figure 7 are initiated by a  
14           block 200 which indicates that the host has to write an entry  
15           into the control queue 131 in the control memory 38 in the SAR  
16           sub-system 29. A block 202 is then activated in Figure 7.  
17           This block provides an indication of the current host position  
18           in the control queue 131 - in other words, where the host is  
19           writing into the control queue in the control memory 38. In  
20           the block 202, the host also reads READ-UD. See the pointer  
21           150 in Figure 5 and 6 and the discussion relating to the  
22           pointer 150.

23  
24           In the next entry, the current position of the host  
25           in the control queue 131 in the control memory 38 is  
26           incremented by an integer. This is indicated at 204 in Figure  
27           7. If this current position is the same as the last known  
28           address written by the SAR sub-system 29 across the PCI bus  
29           130 to the host to indicate where the SAR is in the control  
30           queue 131, the host thinks that the control queue in the  
31           control memory 38 is full. This is indicated at 206 in Figure  
32           7. As indicated at 208 in Figure 7, the host then exits out

1 of the routine of continuing to write to the control queue in  
2 the control memory 38.

3  
4 If the current address of the host in the control  
5 queue 131 in the control memory 38 is not the same as the last  
6 known address in the control queue as seen by the SAR, an  
7 indication is provided at 210. The host then writes across  
8 the PCI bus 130 into the control queue 131 in the control  
9 memory 38 with valid = 1. This is indicated at 212 in Figure  
10 7. As the host writes across the PCI bus 130 into the control  
11 queue 131 in the control memory 38, the host pointer 152 to  
12 the control queue 131 in the control memory 38 is incremented.  
13 This is indicated at 214 in Figure 7. When the writing from  
14 the host to the control queue 131 in the control memory 38 is  
15 completed, an exit is provided as indicated at 216 in Figure  
16 7.

17  
18 Figure 8 is a flow diagram indicating how the SAR  
19 sub-system 29 processes entries from the control queue 131.  
20 In reading the successive blocks in Figure 8, reference should  
21 be made to the table shown in Figure 11 which provides a  
22 definition of the different terms shown in the blocks in  
23 Figure 8. These definitions are supplemented by the previous  
24 discussion relating to Figures 5 and 6 and by the subsequent  
25 discussion relating to Figure 8.

26  
27 As a first step as indicated at 240 in Figure 8, the  
28 SAR sub-system 29 accesses the control queue 131 in the  
29 control memory 38. As indicated at 242, the SAR sub-system 29  
30 then reads the current position of the SAR in the control  
31 queue 131 in the control memory 38. It also counts the number  
32 of times that the processing steps shown in broken lines 258

1 in Figure 8 have been performed. The SAR sub-system 29 then  
2 reads the control information that the host has written into  
3 the control memory 38. This is indicated at 244 in Figure 8.

4  
5 Valid is a bit in the control queue 131 in the  
6 control memory 38. A test is made as at 246 in Figure 8 to  
7 determine whether the valid bit is a binary 1 or a binary 0.  
8 If the bit value is not a binary 1, this indicates that the  
9 entry in the control queue 131 in the control memory 38 is not  
10 valid. Under such circumstances, an exit occurs as indicated  
11 at 248 and access occurs again at the access queue 240 at a  
12 subsequent time. If the valid bit is a binary 1, an  
13 indication is provided on a line 252 in Figure 8. The control  
14 entry is then processed as at 254. In accordance with this  
15 control entry processing, the SAR sub-system 29 processes  
16 control information in the control queue 131 in the control  
17 memory 38.

18  
19 The control information in the control queue 131 in  
20 the control memory 38 tells the SAR sub-system 29 the  
21 addresses of the buffers 134a, 134b and 134c. After the SAR  
22 has processed this information, the VLD (valid) entry in the  
23 control queue becomes 0. This is indicated at 256 in Figure  
24 8. The SAR sub-system 29 then periodically writes to the host  
25 its position in the control queue 131 in the control memory  
26 38. This periodicity may occur once in every fixed number  
27 (e.g. 5) of processing cycles. This periodic processing is  
28 shown within the rectangle 258 defined by broken lines in  
29 Figure 8.

30  
31 In one sense, it is desirable that the fixed number  
32 (e.g. 5) of cycles should be large to minimize the amount of

1 time that the PCI bus is tied up by the transfer of  
2 information across the PCI bus 130 from the host to the SAR  
3 sub-system 29. On the other hand, increasing the fixed number  
4 of cycles is disadvantageous because it decreases the accuracy  
5 of the information in the period of time between the  
6 successive number of processing cycles until the next fixed  
7 number (e.g. 5) of processing cycles has occurred.

8  
9 As a first step in the processing cycles shown  
10 within the broken block 258 in Figure 8, the number of  
11 processing cycles is indicated as at 266 since the last  
12 occurrence of the fixed number (e.g. 5) of processing cycles.  
13 As indicated at 262, the sub-system 29 reads the current SAR  
14 position in the control queue 131 in the control memory 38 and  
15 provides the host with this information in every fixed number  
16 (e.g. 5) of the processing cycles. The count of the  
17 processing cycles is then returned to zero (0) as indicated at  
18 264 in Figure 8, when the fixed number (e.g. 5) of the  
19 processing cycles has occurred.

20  
21 When the count of the number of the processing  
22 cycles is other than the fixed number (e.g. 5), the count is  
23 incremented upon the occurrence of each successive processing  
24 cycle. This is indicated at 269 in Figure 8. Upon each  
25 increment in the count as indicated in the blocks 269 and 266,  
26 the SAR sub-system 29 increments its pointer in the control  
27 queue 131 in the control memory 38 and stores this  
28 information. This is indicated at 270 in Figure 8. When the  
29 processing of the data cells has been completed, an exit is  
30 provided as at 272.

1                Figures 9 and 10 provide a flow chart indicating the  
2        status queue process during segmentation and re-assembly.  
3        Figure 9 relates to the status queue processing of SAR flow  
4        and Figure 10 relates to the status queue processing of host  
5        flow. In reading the successive blocks in Figure 9, reference  
6        should be made to the table shown in Figure 12, which provides  
7        a definition of the different terms shown in the blocks in  
8        Figure 9. These definitions are supplemented by the previous  
9        discussion relating to Figures 5 and 6 and by the subsequent  
10       discussion relating to Figures 9 and 10.

11  
12                As a first step in Figure 9, the SAR sub-system 29  
13        needs to write to the status queue 132. This is indicated at  
14        340 in Figure 9. The SAR sub-system 29 checks to see if the  
15        status queue 132 is full as indicated at 342 in Figure 9. A  
16        binary bit of 1 may be provided if the status queue 132 is  
17        full. If the status queue 132 is full, the SAR sub-system 29  
18        exits as indicated at 343 and performs no further work except  
19        for returning to the block 340 at a subsequent time.

20  
21                If the status queue 132 in the host is not full, the  
22        SAR sub-system 29 indicates the last known position in the  
23        status queue as seen by such SAR sub-system. This is  
24        indicated by the READ\_UD pointer 162 in Figure 6 and is  
25        defined by the READ\_UD variable in the base table (Figure 12).  
26        The SAR sub-system 29 also reads the current position 164  
27        (Figures 5 and 6) of the SAR into the status queue 132. This  
28        is indicated by the WRITE variable in the base table (Figure  
29        12). These two (2) indications provide information to the SAR  
30        sub-system as to whether the status queue 132 is full.

1           The WRITE pointer 164 (Figures 5 and 6) is then  
2   incremented by a count of a binary 1. The incremented  
3   position of the pointer 164 in the status queue 132 is then  
4   compared with the last known host position in the status queue  
5   as seen by the SAR sub-system 29. This comparison is  
6   indicated at 346. If the comparison indicates an equality,  
7   this provides an indication that the status queue is full. A  
8   binary indication is then provided internally to indicate that  
9   the status queue 132 is full. This is indicated at 348 in  
10   Figure 9.

11  
12           A binary 1 is written into the status queue 132 as  
13   indicated at 350 in Figure 9. This indicates that the status  
14   queue entry contains valid information that the host can  
15   process. The current position of the SAR in the status queue  
16   132 is then incremented and the incremented position is  
17   stored. This is indicated at 352 in Figure 9. An exit is  
18   then made as at 354.

19  
20           Figure 10 is a flow diagram showing host entry flow  
21   into the status queue 132. In reading the successive blocks  
22   in Figure 10, reference should be made to the table shown in  
23   Figure 12, which provides a definition of the different terms  
24   shown in Figure 12. These definitions are supplemented by the  
25   previous discussion relating to Figures 5 and 6 and the  
26   subsequent discussion relating to Figure 10.

27  
28           As a first step, the host provides a poll interval  
29   as indicated at 360 in Figure 10. This is a specified period  
30   of time that is provided by the host and entered periodically  
31   by the host. The host then reads the current host position in  
32   the status queue 132 and the number of entries into the status

1 queue 132 since the last writing into the status queue of the  
2 host position in the SAR as seen by the SAR sub-system 29.  
3 This is indicated at 362.

4  
5 As indicated at 364, the host then reads the next  
6 entry that the SAR sub-system 29 writes to the host. The host  
7 then determines whether valid = 1 as indicated at 366 in  
8 Figure 10. A no answer indicates that the entry into the  
9 status queue is not valid. If the answer is no, the host  
10 exits as indicated at 368 and waits for the next poll  
11 interval. If the answer is yes, the host processes entries in  
12 the status queue 132 from the SAR sub-system 29. This is  
13 indicated at 370 in Figure 10. After the host has processed  
14 such entries in the status queue 132, the host writes a valid  
15 = 0 in such entries as indicated at 372 in Figure 10.

16  
17 A determination is then made, as indicated at 374,  
18 whether a fixed number (e.g. 5) of processing cycles has  
19 occurred. The determination of the fixed number (e.g. 5) of  
20 the processing cycles is made in blocks within a rectangle 375  
21 with broken lines. If the answer is yes as indicated at 376  
22 in Figure 10, the host writes across the PCI bus 130 to the  
23 SAR sub-system 29 to inform the SAR system of its position  
24 into the status queue since the last time that it provided the  
25 SAR sub-system such information. This writing is provided by  
26 the READ\_UD Pointer 150 in Figure 5. The host does this only  
27 once in every fixed number of processing cycles to minimize  
28 the amount of time that it ties up the PCI bus. This writing  
29 across the PCI bus is indicated at 378 in Figure 10. The host  
30 then returns the count of the number of processing cycles to  
31 zero (0) to initiate a new count of the number of the  
32



1 processing cycles to the fixed number (e.g. 5). This is  
2 indicated at 380 in Figure 10.

3  
4 If the count of the number of processing cycles is  
5 not equal to the fixed number (e.g. 5), an indication is  
6 provided on a line 384 in Figure 10. This causes the number  
7 of cycles to be incremented by one (1) every time that the  
8 number of the queue entries in the status queue 132 is updated  
9 since the last time that the host has informed the SAR sub-  
10 system 29 what it has read out. This is indicated at 386 in  
11 Figure 10. The read pointer indicating the host position in  
12 the status queue is then incremented as indicated at 388.  
13 This occurs whether the count of the successive processing  
14 cycles is at the fixed number and is being returned to zero  
15 (0) as indicated at 380 or whether the count of the processing  
16 cycles is being incremented toward the fixed number (e.g. 5)  
17 as indicated at 386.

18  
19 Figure 11 is a table defining certain terms used in  
20 the blocks shown in Figures 7 and 8. The definitions included  
21 in the table shown in Figure 11 may be considered to  
22 supplement the definitions provided in the specification.  
23 Figure 12 is a table defining certain terms used in the blocks  
24 shown in Figures 9 and 10. The definitions included in the  
25 table shown in Figure 12 may be considered to supplement the  
26 definitions provided in the specification.

27  
28 Although this invention has been disclosed and  
29 illustrated with reference to particular embodiments, the  
30 principles involved are susceptible for use in numerous other  
31 embodiments which will be apparent to persons of ordinary  
32

skill in the art. The invention is, therefore, to be limited  
only as indicated by the scope of the appended claims.

What is claimed is: